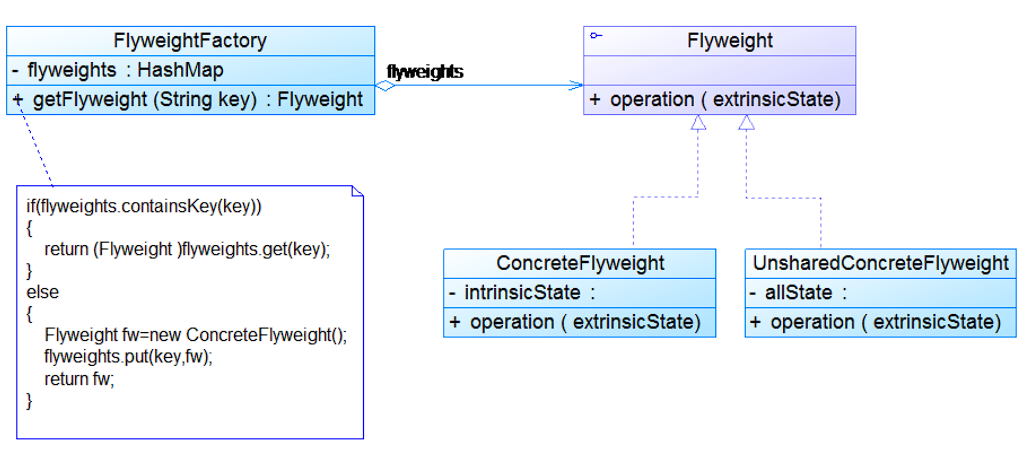
1. 模式定义

享元模式（FlyweightPattern)：运用共享技术有效地支持大量细粒度对象的复用。系统只使用少量的对象，而这些对象都很相似，状态变化很小，可以实现对象的多次复用。由于享元模式要求能够共享的对象必须是细粒度对象，因此它又称为轻量级模式，它是一种对象结构型。

模式。

**二、**模式动机

在享元模式中可以共享的相同内容称为内部状态，而那些需要外部环境来设置的不能共享的内容称为外部状态，由于区分了内部状态和外部状态，因此可以通过设置不同的外部状态使得相同的对象可以具有一些不同的特征，而相同的内部状态是可以共享的。在享元模式中通常会出现工厂模式，需要创建一个享元工厂来负责维护一个享元池用于存储具有相同内部状态的享元对象。

在享元模式中共享的是享元对象的内部状态，外部状态需要通过环境来设置。在实际使用中，能够共享的内部状态是有限的，因此享元对象一般都设计为较小的对象，它所包含的内部状态较少，这种对象也称为细粒度对象。享元模式的目的就是使用共享技术来实现大量细粒度对象的复用。

享元模式是一个考虑系统性能的设计模式，通过使用享元模式可以节约内存空间，提高系统的性能。

三、例子

编辑器软件，例如多处使用同一张图片。

四、优缺点

优点：

·享元模式的优点在于它可以极大减少内存中对象的数量，使得相同对象或相似对象在内存中只保存一份。

·享元模式的外部状态相对独立，而且不会影响其内部状态，从而使得享元对象可以在不同的环境中被共享。

缺点：

·享元模式使得系统更加复杂，需要分离出内部状态和外部状态，这使得程序的逻辑复杂化。

·为了使对象可以共享，享元模式需要将享元对象的状态外部化，而读取外部状态使得运行时间变长。

五、适用环境

·一个系统有大量相同或者相似的对象，由于这类对象的大量使用，造成内存的大量耗费。

·对象的大部分状态都可以外部化，可以将这些外部状态传入对象中。

·使用享元模式需要维护一个存储享元对象的享元池，而这需要耗费资源，因此，应当在多次重复使用享元对象时才值得使用享元模式。

六、示例代码

